基于 FPGA 的频域幸运成像算法

黄学明 [1]，李彬华 [1,2]，王锦良 [1]

（1. 昆明理工大学信息工程与自动化学院，云南 昆明 650500；

2. 昆明理工大学云南省计算机应用技术重点实验室，云南 昆明 650500）

摘 要：幸运成像技术是一种事后图像恢复技术，它能够有效减小大气湍流对图像质量的影响。传统的频域幸运成像算法在效果上优于传统的空域算法，它主要由图像预处理、数据选择和数据叠加三部分组成。本文提出了一种基于现场可编程门阵列（FPGA）的频域幸运成像算法，并在 FPGA 平台上构建了频域幸运成像的实验系统。本系统通过二维转换为两个一维实现的处理方式对天文图像数据进行了二维快速傅里叶变换（2D-FFT），采用频域数据的实部平方与虚部平方之和代替幅值进行数据选择，所得到的高分辨率图像与基于中央处理器（CPU）算法处理的结果基本相同。本实验系统能对 2000 帧 128×128 像素的图像进行频域幸运成像处理，比传统的基于 CPU 频域幸运成像算法速度快 13 倍多，实现了对传统的频域幸运成像算法进行加速，同时也为将来频域幸运成像的实时化提供了一种可行的技术方案。

关键词：幸运成像；大气湍流；2D-FFT；频域；FPGA

中图分类号：TP391.4

大气湍流是大气中一种不规则的随机运动，它使望远镜的实际分辨率远低于望远镜的衍射极限分辨率，它是限制地基光学望远镜空间分辨率的一个主要因素[1]。地面观测要获得接近望远镜衍射极限分辨率的图像，可以使用事后图像恢复或重建技术，而幸运成像技术就是一种用于去除大气湍流的事后图像恢复技术[2,3]。幸运成像技术既可以在空域上进行，也可以在频域上进行，但是空域算法的选图主要是以帧为单位对图像进行选取，这样的选图方式并没有考虑到图像数据在某些方向的高频分量信息，在选取数据处理时，会导致图像数据信息的浪费，因此，在 2012 年，Garrel 等人提出了一种基于傅里叶变换的幸运成像算法，该算法主要由图像预处理、数据选择和数据叠加三部分组成，并对提出的方法进行了模拟实验，实验结果表明，此算法以 10%的数据选择率进行数据选择时，成像效果最佳**错误!未找到引用源。**。2013 年 Mackay 以实测的数据进一步证实了频域幸运成像的有效性，同时也指出了该算法相对于空域算法在计算量上存在缺陷**错误!未找到引用源。**。2019 年本实验室胡兴等人对传统的频域幸运成像算法进行改进，在频域上采取了分组数据选择的方法对传统的频域幸运成像算法进行加速，但该算法仍然属于事后处理的范畴**错误!未找到引用源。**。

上述频域幸运成像算法都是基于 CPU 的，因其串行的处理方式，难以实时处理，使观测人员无法在天文观测过程中准确地了解观测天体信息，因此，实时化是幸运成像技术发展的必然趋势。近些年来，两大并行的硬件加速器引入了图像处理领域[6-7]，一个是图形处理器（Graphics Processing Unit，GPU），另一个是 FPGA。虽然两者都具有强大的并行处理能力，但是 FPGA 是一种半定制型电路，可以根据实际需求去编程，而 GPU 一旦设计完成后就固定了，因此，GPU 的灵活度远不如 FPGA[8]。故本文选择 FPGA 来实现频域幸运成像算法的加速。

本文根据 FPGA 强大的并行性和灵活性，提出了一个基于 FPGA 的频域幸运成像算法，然后按 FPGA 并行和流水线设计方法，对该频域幸运成像算法用 Verilog 硬件描述语言（Hardware Description Language, HDL）进行了设计，并在 Xilinx Spartan-7 上进行了工程实现和相关测试实验。本文分 4 节，其中第 1 节描述算法，第 2 节介绍该算法在 Spartan-7 FPGA 上的系统设计过程，第 3 节是实验结果和讨论，第 4 节是结论。

1 适用于 FPGA 的频域幸运成像算法

传统的频域幸运成像算法主要由图像预处理、频域内数据选择和数据叠加三部分组成，在频域内选择数据时，需要消耗大量的时钟资源，导致算法运行速度非常缓慢，目前只有胡兴对此问题探讨过，但还是属于事后处理的范畴。本节将提出一个新的频域幸运成像算法，并在 FPGA 上进行加速，且在台式机上验证了算法的有效性。

1.1 算法的总体设计

本文提出的基于 FPGA 的频域幸运成像算法既不同于文献[3]和文献[4]的传统频域幸运成像算法，也不同于文献[5]的改进的频域幸运成像算法。在文献[4]中，Carrel 等人提出在频域内以幅值为基准对图像数据进行选择；在文献[3]中，Mackay 在证实了频域幸运成像有效性的同时，也提出了该算法计算量非常大的缺陷；在文献[5]中，胡兴用分组的方法对频域幸运成像算法进行了加速，但处理速度仍然非常缓慢。因此，将频域幸运成像技术在 FPGA 上处理是该算法加快处理速度的有效途径。

在 FPGA 上进行频域幸运成像有三个技术难点：

第一，在 FPGA 上无法对图像数据直接进行二维快速傅里叶变换。图像数据在 FPGA 上做二维傅里叶变换与在 MATLAB 平台完全不一样，在 MATLAB 平台可以直接对图像数据做二维变换，而在 FPGA 上只能做一维傅里叶变换。因此，根据二维傅里叶变换的原理，若有一个 N 行 M 列的空域图像数据 f(x,y)，其二维傅里叶变换后的频率点数据可为F(u,v)，此F(u,v)为复数，定义式如（1）式所示：

$$F(u,v)=\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}f(x,y)e^{-j2\pi(ux/M+vy/N)} \qquad (1)$$

对定义式（1）进行分解，即得到定义式如（2）式所示：

$$F(u,v)=\sum_{x=0}^{M-1}(\sum_{y=0}^{N-1}f(x,y)e^{-j2\pi(vy/N)})e^{-j2\pi(ux/M)} \qquad (2)$$

式中 x，y 为空域图像的行数和列数，其取值范围为 x=0，1，…，M-1；y=0，1，…，N-1；式中 u，v 为频域图像的行数和列数，其取值范围为 u=0，1，…，M-1；v=0，1，…，N-1。由定义式（1）（2）可知，二维傅里叶变换可采用二维转换为两个一维实现的算法进行处理，即将 2D-FFT 拆分成行方向的 FFT 和列方向的 FFT，然后按先后次序进行计算。通常而言，一维 FFT 算法的行列先后顺序对 2D-FFT 运算结果没有影响，且由于本系统是对 128×128 大小的图像进行二维快速傅里叶变换，故 N，M 都等于 128。

第二，传统的频域幸运成像算法是以幅值为基准来选择数据的，即按式（3）选择数据。

$$M(u,v)=\sqrt{\mathrm{Re}[F(u,v)]^2+\mathrm{Im}[F(u,v)]^2} \qquad (3)$$

式中 Re[F(u,v)]为频域数据的实部，Im[F(u,v)]为频域数据的虚部，M(u,v)是频域数据的模。

虽然在 FPGA 上可以方便的进行整数的加减乘除运算，但是在 FPGA 上计算复数幅值的算法复杂且耗时，除非调用 Cordic 知识产权核（International Property，ip），但是此 ip 核将消耗大量的硬件资源，不适合本系统在中小规模的 FPGA 上实现。因此，本文提出使用频域数据的实部平方与虚部平方之和代替幅值进行数据选择，因为实部的平方与虚部的平方之和与幅值的变化规律是一致的，即使用式（4）代替式（3）进行数据选择。

$$M^2(u,v)=\mathrm{Re}[F(u,v)]^2+\mathrm{Im}[F(u,v)]^2 \qquad (4)$$

第三，FPGA 片上存储资源有限。由于本系统需要对大量的图像频域数据进行存储，而

片上 RAM 资源远远不足，因此，本系统使用 FPGA 片外大容量存储器进行频域图像数据的存储。

本文提出的适用于 FPGA 的频域幸运成像算法大致工作流程如下所述：

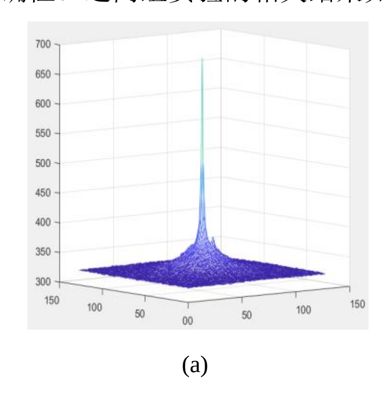
(1) 图像输入，并进行高斯滤波；
(2) 滤波后的图像做行快速傅里叶变换；
(3) 行傅里叶变换的结果用乒乓操作的方式在 RAM 中进行转置；
(4) 从 RAM 中读出数据做列快速傅里叶变换，并将结果存入 DDR3；
(5) 从 DDR3 中读出数据，按 10%的比例选择数据；
(6) 数据叠加；
(7) 对叠加数据做行快速傅里叶反变换；
(8) 行傅里叶反变换的结果在 RAM 中进行转置；
(9) 从 RAM 中读出数据做列快速傅里叶反变换；
(10) 将列傅里叶反变换的结果分成两路，一路数据进行二值化，并送入 HDMI 显示模块进行显示，另一路数回传到上位机。

在以上工作流程中，由于 FPGA 具有强大的并行处理能力，第(1)步至第(4)步是并行处理的，第(5)步至第(8)步也是并行处理的。第(1)步图像数据是通过精简吉比特介质独立接口(Reduced Gigabit Medium Independent Interface，RGMII)输入的，第(2)步至第(4)步是傅里叶正变换过程，即图像数据从空域变换到频域的过程，第(7)步至第(9)步是傅里叶反变换过程，即图像数据从频域变换到空域的过程。

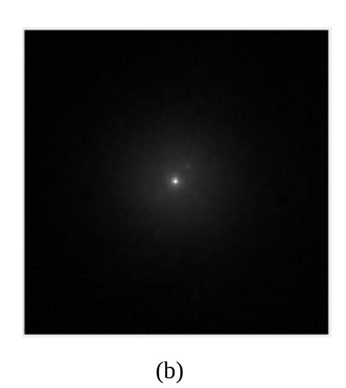
1.2 算法的验证

为了验证该频域幸运成像算法的可行性，同时为了验证该频域幸运成像重建的高分辨率图像比空域幸运成像重建的高分辨率图像效果更佳，本文在 CPU+MATLAB 平台分别做了两组实验，第一组是以幅值为基准选择数据的传统频域算法，第二组实验是在 MATLAB 上模拟本文提出的算法在 FPGA 上的实现，并与文献[10]的实验结果进行对比分析。本实验的硬件平台是 Dell Precision T5500 图像工作站，内存 16GB，CPU 为 Xeon E5620，显卡为 NVIDIA GTX1080，运行的软件环境是 Windows 7 操作系统、MATLAB R2017a。

在 MATLAB 中对随机抽取的 2000 帧128×128像素大小的图像进行频域幸运成像算法处理，根据 Garrel 等人及胡兴等人的研究结果显示，数据选择的比例取 10%时，频域幸运成像的效果是最好的。第一组实验是对 Garrel 等人提出的传统频域算法进行复现，其正确性已经过多次验证，本文将不再详细介绍；第二组实验是验证本文提出的频域幸运成像算法的正确性。这两组实验的相关结果如图 1、2 所示。



(a)



(b)

变量 - sum_u_f

sum_u_f

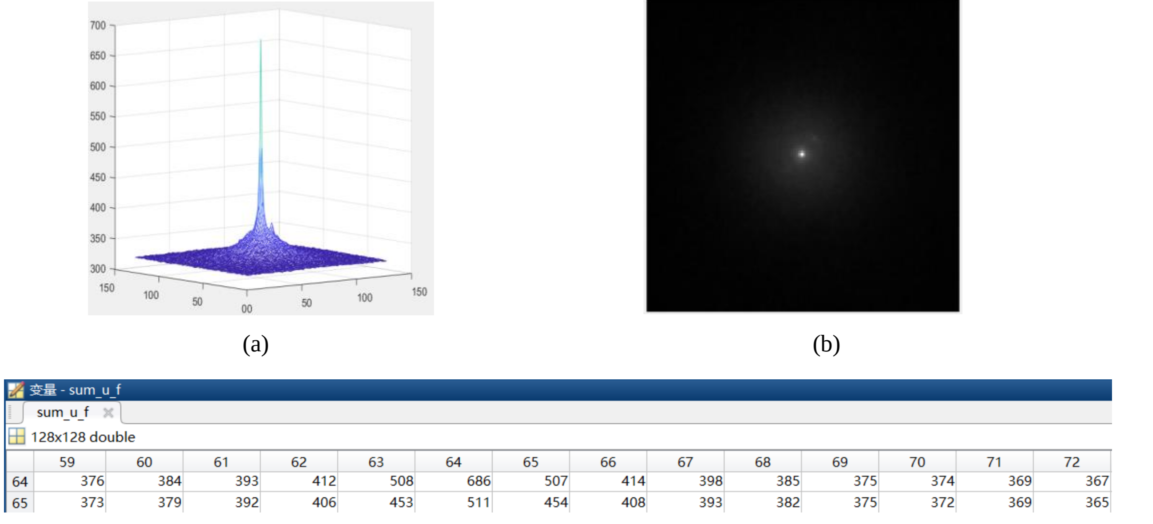
128x128 double

	59	60	61	62	63	64	65	66	67	68	69	70	71	72
64	376.4095	384.0014	393.9555	412.9262	508.8520	686.7111	507.2008	414.3027	398.2838	385.9003	375.9284	374.5754	369.6463	367.3767
65	373.9176	379.3518	392.7328	406.6559	453.9716	511.0316	454.9955	408.6455	393.8191	382.8735	375.2743	372.4034	369.3520	365.6716

(c)

图 1 传统算法的结果。(a)三维灰度分布图; (b)二维灰度图; (c) 峰值数据

Fig.1 The result of traditional algorithm. (a)3D gray distribution map; (b)2D gray image; (c)Peak data



(a) (b)

	59	60	61	62	63	64	65	66	67	68	69	70	71	72
64	376	384	393	412	508	686	507	414	398	385	375	374	369	367
65	373	379	392	406	453	511	454	408	393	382	375	372	369	365

(c)

图 2 本文提出算法的结果。(a)三维灰度分布图; (b)二维灰度图; (c) 峰值数据

Fig.2 The result of the paper’s algorithm. (a)3D gray distribution map; (b)2D gray image; (c)Peak data

根据上述的实验结果，从直观上看不出有任何误差，然后对两组实验的数据进行分析，如图 1(c)、图 2(c)所示，由于第二组实验是模拟算法在 FPGA 上的实现，所以整个运算过程都是对整数进行处理，从图中数据可以得出，本文提出的适用于 FPGA 的频域算法跟传统的算法仅在小数部分存在误差，即截断误差，这是由于 FPGA 只对整数进行处理造成的。同时，为了进一步分析这两组频域算法重建高分辨率结果图的优劣，我们也选择了 3 种客观的图像质量评价函数，通过数值计算进行对比分析，其中 FWHM 为天文图像常用评价指标，数值越小，说明图像质量越好，其他 2 种评价函数均为经典的数字图像清晰度评价指标，数值越大，说明图像质量越好。表 1 中的数据是三种算法分别对 2000 帧图像处理得到的高分辨率结果图像的图像质量评价指标值，这三种算法分别是上述两组频域幸运成像算法和文献[10]空域幸运成像算法。

表 1 两组频域算法和空域算法的图像质量评价结果

Table1 The evaluation results of image quality

Evaluation functions	Traditional algorithm	The paper’s algorithm	Reference[10]
FWHM	0.188"	0.188"	0.215”
Tenengrad gradient	29.03	27.86	13.95
Laplacia gradient	19.63	19.61	11.34

从表 1 中可以看出，本文提出的频域算法与传统的频域算法相比，FWHM 的值是一样的，Tenengrad 梯度和 Laplacia 梯度值略低于传统的频域算法，这都是 FPGA 运算过程中产生的截断误差造成的，说明本适用于 FPGA 的频域算法得到的高分辨率图像质量比传统算法略差，但是，从表 1 中本文提出的频域算法数据和文献[10]空域系统的数据对比可知，本文提出的频域算法得到的图像质量（清晰度）远比空域算法得到的图像质量要好，也说明本文提出的频域算法是可行且有效的。

既然所提算法是可行且有效的，那么只要 FPGA 设计合理，就可以缩短处理时间，使频域幸运成像在 FPGA 上进行加速，并为频域幸运成像的实时化提供一种可行的技术方案。所以，在完成上述算法设计后，频域幸运成像系统的 FPGA 实现就成为下一个关键问题。

2 频域幸运成像算法的 FPGA 设计及系统实现

算法的设计及实现的系统通常都与所用硬件平台相关。本文的频域幸运成像算法的硬件实现平台是以 Xilinx 公司 Spartan-7 系列的 XC7S50 芯片为核心的开发板和一台 CPU 为 Xeon E5620 的工作站，设计开发环境是 Vivado2019.1，采用 Vivado 平台上的内嵌式逻辑分析仪和 ModelSim SE 10.7 进行硬件设计的验证与调试。在本系统的研究过程中，将前面提出的算法用 Verilog HDL 进行了逻辑设计，并在 FPGA 硬件上实现。本文将重点介绍系统的总体设计以及 FFT_IFFT 模块、数据选择模块的算法及其 FPGA 实现。下面分 3 小节来介绍具体的设计与实现过程。

2.1 频域幸运成像系统的总体设计

本系统主要包含了三大部分，即数据的传输、数据的运算及数据的存储。由于系统将来要面向的是实时应用，对数据的传输速度要求比较高，因此，采用了千兆以太网进行数据传输；在数据的运算方面，主要涉及到了二维快速傅里叶变换，而在 FPGA 上无法对图像数据直接做二维快速傅里叶变换，故本文使用了二维转换为两个一维实现的方式对天文图像数据做二维快速傅里叶变换；在数据的存储方面，使用了第三代双倍数据率同步动态随机存取存储器（Double-Data-Rate Three Synchronous Dynamic Random Access Memory，DDR3），因为本文是对 2000 帧 128×128 像素的图像进行处理，而 FPGA 的片上 RAM 资源无法满足本系统的存储要求，故采用了片外 DDR3 资源对数据进行储存。并用 Verilog HDL 对系统设计进行实现。系统的总体结构框图如图 3 所示。

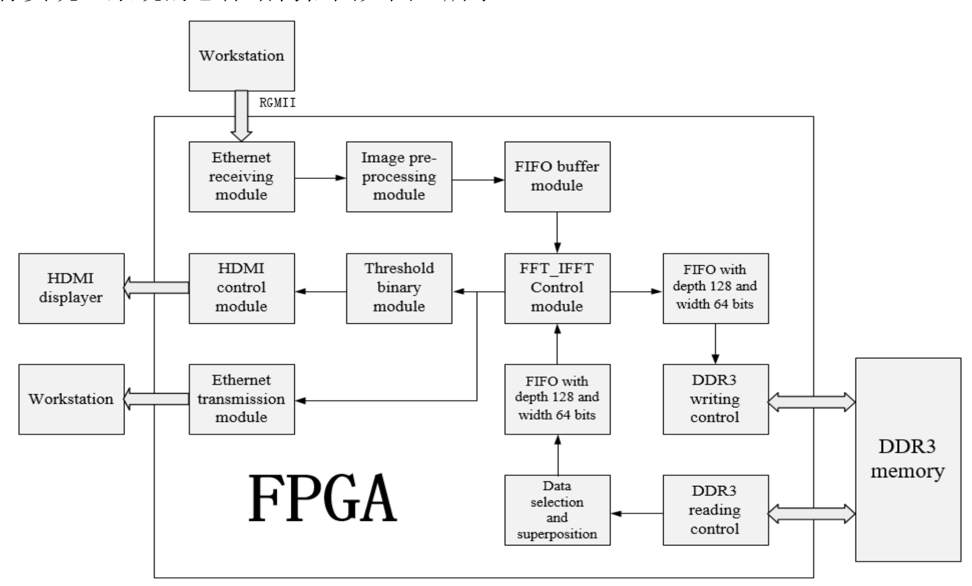


图 3 本系统的总体结构框图

Fig3. Overall structure of the system

由于受到了本系统所使用的 FPGA 硬件平台片上 RAM 资源的限制，目前只能使用 2000 帧 128×128 像素的短曝光图像作为原始图像进行频域幸运成像处理，并最终以 128×128 像素大小显示出频域幸运成像的结果。

2.2 FFT_IFFT 算法及其实现模块设计

传统的空域幸运成像算法是在空域上进行选图和叠加，而频域幸运成像算法则是在傅里

叶图像的每个频率点上进行数据选择和叠加，因此，要想实现频域幸运成像，必须将图像数据做傅里叶变换，并在频域上进行数据的选择和叠加。Xilinx 公司在 Vivado 平台上提供了一维快速傅里叶变换的 ip 核，但由于图像数据是二维的，因此并不能直接使用一维 FFT ip 对空域图像数据进行运算，因此，本模块使用了二维转换为两个一维实现的方式对空域图像做二维傅里叶变换。本系统的 FFT_IFFT 模块设计结构框图如图 4 所示。本模块的工作流程描述如下：

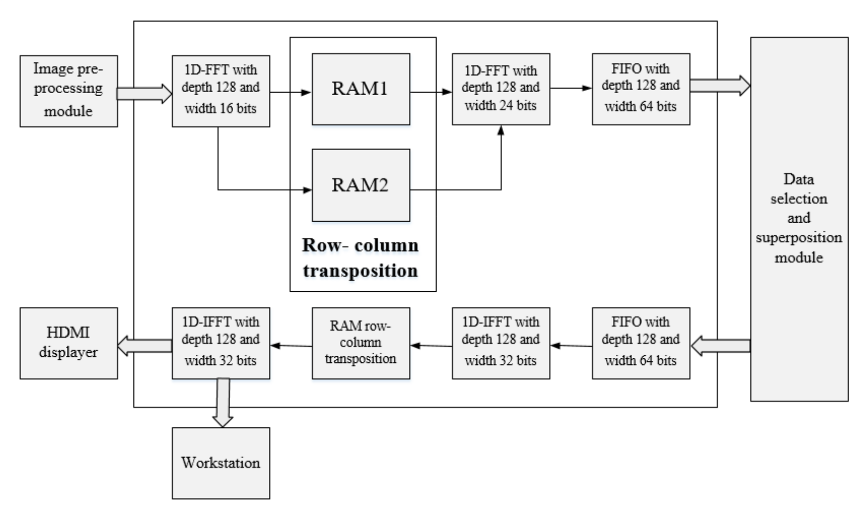


图 4　2D-FFT 模块的结构框图

Fig4.　The structure of 2D-FFT module

(1) 将预处理模块处理后的数据先进行 FFT 行变换，再将行变换的结果在 RAM1，RAM2 中进行转置，这里使用了两个深度 16384，宽度 48 位的简单双端口 RAM 进行乒乓操作（本系统原始图像数据为 16 位，经一次傅里叶正变换后，得到的实部和虚部数据各 24 位，故这里使用宽度 48 位的 RAM 进行转置操作），即当 RAM1 写满 1 帧图像数据时，RAM1 进行读操作，读出的数据做 FFT 列变换，此时 RAM2 进行写操作；当 RAM2 写满 1 帧图像数据时，RAM2 进行读操作，读出的数据做 FFT 列变换，此时 RAM1 进行写操作。最后再将行列变换后的频域数据存入 DDR3 中进行数据的选择和叠加。

(2) 将选择叠加后的数据先进行行 IFFT，再将逆变换的结果存入 RAM 中，这里使用了一个深度 16384，宽度 64 位的简单双端口 RAM 进行转置，即当 RAM 存完行快速傅里叶逆变换的结果后，再开始进行列 IFFT，即得到叠加后的空域图像数据，并将此数据分为两路，一路数据求平均，然后进行重建高分辨率图像的显示；另一路经以太网发送模块将数据回传至上位机。

2. 3 数据选择算法及其实现模块设计

传统的数据选方择式是将所有图像相对应的所有频域点的幅值通过排序建立索引关系，选出幅值大小在前 10%的频率点数据，这种数据选择的方式计算量非常大，并且消耗的硬件资源也很多，即使是 Xilinx 推出的高端 Vertex-7 系列 FPGA 也无法满足这种数据选择方式所消耗的硬件资源。因此，第一，本文使用选择数据但不排序的方式进行数据选择，第二，本文使用频域数据的实部平方和虚部平方之和代替幅值选择数据，具体的模块设计结构框图如图 5 所示。数据选择的工作流程如下：

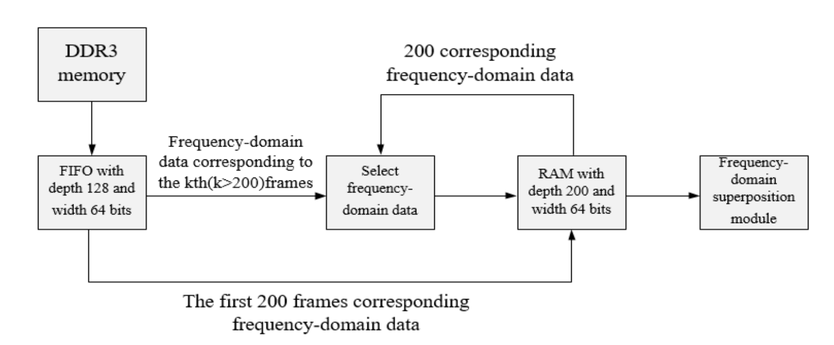


图 5 数据选择模块的结构框图

Fig5. The structure of data selection module

首先，从 DDR3 中读出图像的频域数据，读完第 1 帧图像的第一个数据经 FIFO 存入到 RAM 中后，下一次读出的就是第 2 帧图像相对应的数据（即第一个数据），且以同样的方式存入 RAM 中，一直到第 200 帧相对应的数据存入 RAM 后，即从第 201 帧开始，读出相对应的数据先存入 FIFO 中，并在该数据存入 RAM 之前，要先将 RAM 中前 200 相对应点的数据的实部平方与虚部平方之和进行比较，找出最小的那个数值及地址，再将该任意数据的实部平方与虚部平方之和跟前面找出的最小的数值进行比较，若大于前面找出的最小值，则将该数据存入前面找出最小值的地址，若小于前面找出的最小值，则该数据直接舍去。从第 202 帧到第 2000 帧，一直重复第 201 帧的第一个数据这个过程，当第 2000 帧相对应的数据存完后，RAM 将该 200 个数据送到叠加模块进行叠加。其次，将每帧图像的第 2 个至第 16384 个数据完全重复前述的操作，即完成了所有图像的频域数据选择。

3 实验结果及其对比分析

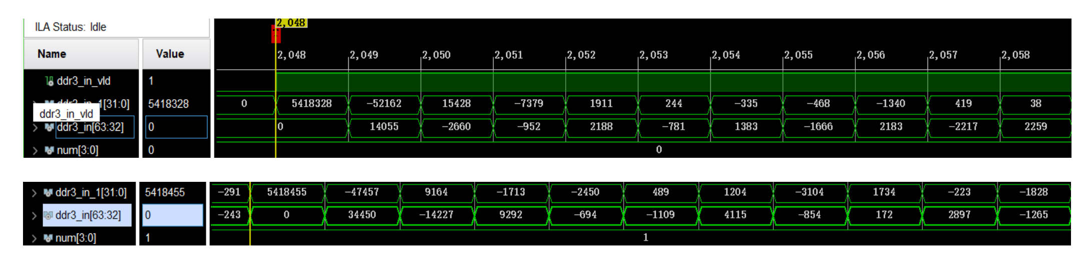
为了验证本文提出的数据选择方法在 FPGA 上的实现是正确的，使用内嵌式逻辑分析仪对数据选择模块的数据进行抓取；为了验证频域幸运成像算法在 FPGA 硬件平台上实现的正确性，本文以同样的算法在 CPU+MATLAB 平台上进行模拟实验，并将此算法在两个平台上处理的结果进行对比分析。

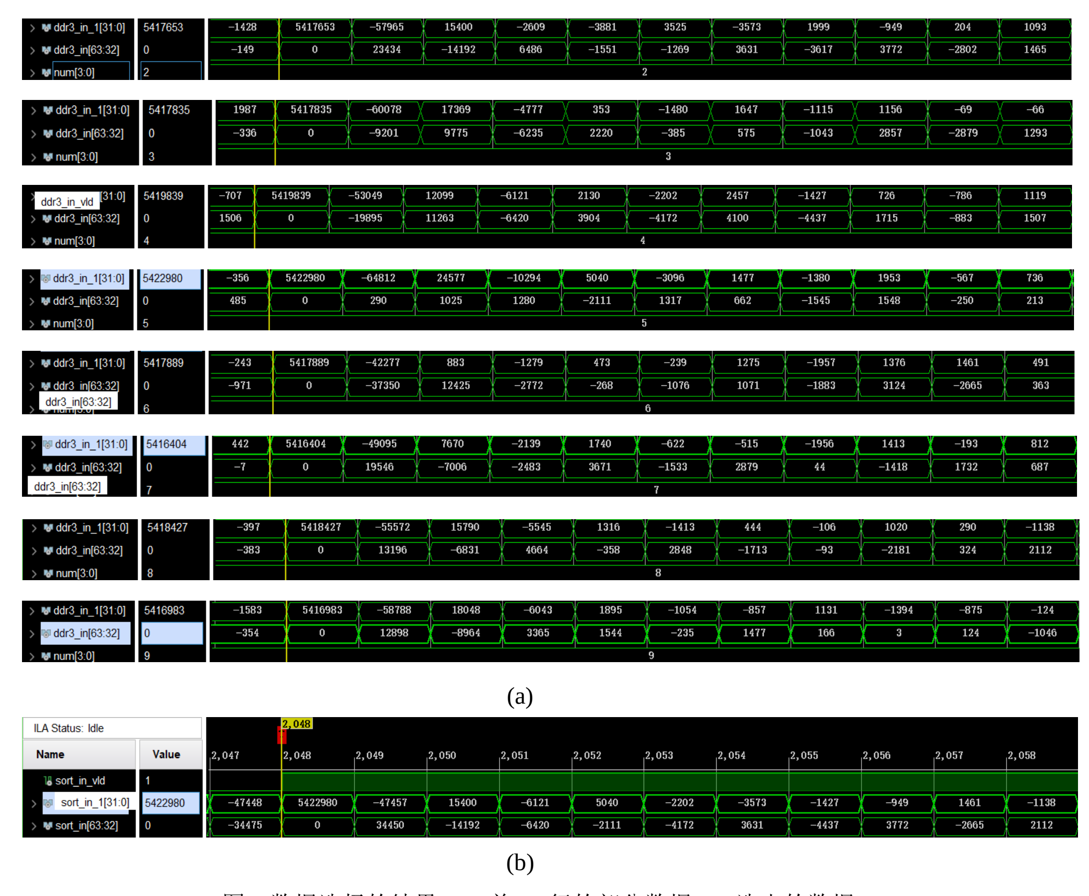
本文采用 2016 年 10 月 20 日在云南天文台丽江观测站对天文双星 HDS 70 的观测图像，共 10000 帧，这组图像的观测条件和参数见文［9］。在图像帧数以及图像大小的选取上，因受 FPGA 硬件资源的限制，多次随机从 10000 帧图像中抽取 2000 帧，并裁剪成 128 × 128 像素大小的天文目标短曝光图像进行处理。

3.1 频域幸运成像在 FPGA 平台上的实验结果及对比分析

3.1.1 数据选择算法结果

本系统采用的是2000帧图像每个相对应的频率点都选出200个频率点的数据选择方法，即 10%的数据选择比例。为了使内嵌逻辑分析仪抓取的数据更加直观，这里截取了前 10 行的部分数据按 10%的比例进行数据选择，即 10 行数据选出 1 行。数据选择结果如图 6 所示。





(a)

(b)

图 6 数据选择的结果。(a)前 10 行的部分数据; (b)选出的数据

Fig.6　The result of the data selection. (a)the part data of the first 10 lines; (b)selected data

从图 6(a)、图 6(b)中可以看出，每行相对应数据的实部平方与虚部平方之和最大的那个数被选出，同样也是幅值最大的那个数被选出，说明内嵌逻辑分析仪抓取的数据结果是正确的，也说明本文提出的数据选择算法是正确的。

3.1.2 频域幸运成像在 FPGA 和 MATLAB 上的结果分析

本系统的硬件平台是以 Xilinx 公司 Spartan-7 系列的 XC7S50 芯片为核心的开发板。本系统基于 FPGA 上实现的频域幸运成像算法的验证，采用 10%的数据选择比例做处理。由于在 MATLAB 和 FPGA 上所得的高分辨率图像不易进行比较，故本系统在 FPGA 和 MATLAB 上均对算法处理所得的高分辨率图像做了相同的锐化，将最终得到的高分辨率图像中感兴趣的目标或区域突出出来，分别得到了如图 7(a)、7(b)所示的锐化图像。

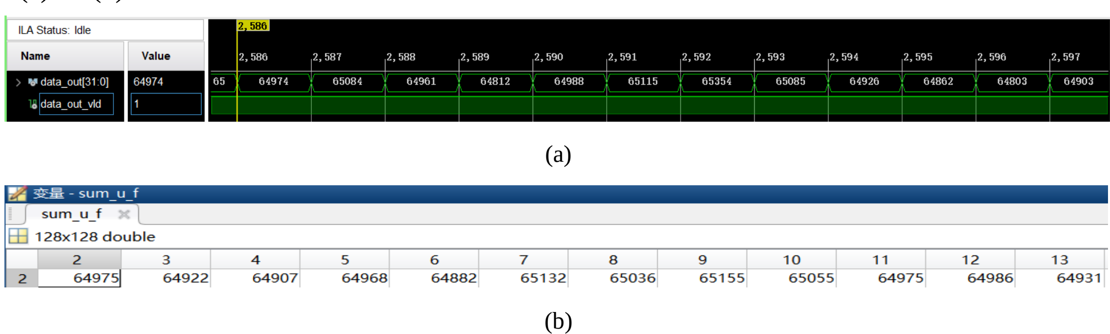


(a)　　　　　　　　　　　　(b)

图 7　频域幸运成像处理结果。(a) FPGA 处理结果; (b) MATLAB 处理结果

Fig.7 Result of frequency-domain lucky imaging.(a)Result of FPGA processing;(b)Result of MATLAB processing

图 7 (a)是由以太网回传到上位机的图像数据经锐化后调用 MATLAB 函数显示的结果图，图 7(b)是在 MATLAB 上进行频域幸运成像算法处理并做相同锐化后的结果图。通过对比，可以看出两帧图像是完全相同的，从而验证了本系统在 FPGA 上实现的频域幸运成像算法是正确的。但是，由于 FPGA 只能处理整数，所以在数据进行快速傅里叶变换时，会产生截断误差，不过该误差非常的小，通常在一个数以内。为了使数据的对比更加的直观，在重建后的高分辨率图像数据做平均之前（即 200 帧图像叠加起来的结果），随机截取了一段 FPGA 处理后的数据（使用内嵌式逻辑分析仪抓取数据）与 MATLAB 处理后的数据进行对比，发现大部分数据误差都在 200 以内，即做完平均后误差基本上都在 1 之内，这说明图像数据在做快速傅里叶变换时产生的截断误差对图像的质量很小，随机截取的数据如图 8(a)，8(b)所示：



(a)

(b)

图 8 频域幸运成像处理结果的部分数据。(a) FPGA 处理结果; (b) MATLAB 处理结果

Fig.8 The part data of frequency-domain lucky imaging results. (a)By FPGA; (b) By MATLAB

3. 2 时钟消耗对比

将本文所提出的基于 FPGA 频域幸运成像算法和文献[10]赵盼孜提出的基于 FPGA 幸运成像算法（空域）对 1000 帧原始图像进行幸运成像实验，就它们的算法运行时间、图像数据读取时间和总运行时间进行对比，实验结果如表 2 所示。

表 2 1000 帧128×128像素大小的天文图像的幸运成像算法运行时间（单位：秒）

Table 2 Running times (in seconds)

Computing system and platform	Time for algorithm	Time for image transmission	Total
ZHAO’s system（FPGA）	2.45	41.94	44.39
The paper’s system（FPGA）	1.94	4.99	6.93

注：由于赵盼孜系统处理能力有限，所以这里只对 1000 帧图像进行实验。

由于频域幸运成像算法处理的数据量远比空域幸运成像要多，因此，频域幸运成像的算法复杂度也比空域幸运成像要高得多，但是从表 2 可以看出，本文所提出的频域幸运成像算法在 FPGA 平台上的处理速度却比赵盼孜的空域系统快了 0.51 秒，这说明本算法充分的利用了 FPGA 的并行性和灵活性优势；本系统像素读取速度比赵盼孜的空域系统像素读取速度快 8.4 倍，这是因为本系统采用千兆以太网传输方式取代了 SD 卡的传输方式；本系统的平均运行速度是赵盼孜空域系统的 6.4 倍，这说明了本算法在 FPGA 实现的合理性与正确性。

4 结论

本文在仔细分析 Garrel 等人提出的一种基于傅里叶变换的幸运成像算法以及胡兴等人提出的一种改进的频域幸运成像算法后，提出了基于 FPGA 的频域幸运成像算法，并将该频域幸运成像算法在 FPGA 上实现了加速。首先，本文对算法进行了详细的概述，并在 MATLAB 上验证了该算法是可行且有效的，同时验证了本文提出的频域算法在成像效果上

要优于空域算法；其次，本文对算法的总体设计以及二维快速傅里叶变换模块，数据选择模块，数据叠加模块的设计思路进行了介绍；最后，本文在一块中小规模低成本的 FPGA 开发板上进行了系统的实现，并展示了相应模块的测试结果与验证过程。由于 MATLAB 的代码效率是比较低的，不太适合用于天文观测，而比较合适用来做算法验证或实验结果分析，因此，本文将该频域系统在 FPGA 平台上的实验结果与 MATLAB 平台上的实验结果进行了对比，证明了该系统在 FPGA 上实现的正确性。虽然本设计方案还存在着不足之处，跟动态实时化相比还有一定的差距，但是就频域幸运成像算法在 FPGA 上的具体实现，并在 FPGA 上实现了加速处理，为频域幸运成像的实时化提供了一种可行的技术方案。

致谢：本实验所用双星 HDS 70 的短曝光图像的原始数据由中国科学院云南天文台张西亮提供，特此致谢。本文研究工作受到中国国家自然科学基金委员会的资助，项目编号 11673009。

参考文献:

[1] OSCOZ A，REBOLO R，LOPEZ R，et al. FastCam: a new lucky imaging instrument for medium-sized telescopes [D]// Proceeding of SPIE，2008.

[2] BALDWIN J E，MACKAY C D，LAW N M. Lucky imaging: high angular resolution imaging in the visible from the ground [J]. Astronomy and astrophysics，2006，446(2) :739-745.

[3] MACKAY C D. High-efficiency lucky imaging [J]. Monthly Notices of the Royal Astronomical Society，2013，432(1) : 702－710.

[4] GARREL V，GUYON O，BAUDOZ P. A Highly Efficient Lucky Imaging Algorithm: Image Synthesis Based on Fourier Amplitude Selection [J]. Publications of the Astronomical Society of the Pacific，2012，124(918) :861-867.

[5] 胡兴. 频域幸运成像算法研究 [D]. 昆明理工大学硕士学位论文，2019，3.
HU X. Research on frequency-domain lucky imaging algorithm[D]，Master's degree thesis of Kunming University of Science and Technology，2019，3.

[6] 郭诚欣，陈红，孙辉，等. 基于现代硬件的并行内存排序方法综述 [J]. 计算机学报，2017，40(7):2070-2092.
GUO C X，CHEN H，SUN H，et al． Parallelism of in-memory sorting algorithm on modern hardware [J].Chinese Journal of Computers，2017，40(9) : 2070－2092.

[7] HEGARTY J，DALY R，DEVITO Z，et al. Rigel: flexible multi-rate image processing hardware [J]. ACM Transactions on Graphics, 2016, 35(04): 85.

[8] STEWART R，DUNCAN K，MICHAELSONG，et al. RIPL: a parallel image processing language for FPGAs [J]. ACM Transactions on Reconfigurable Technology and Systems，2018，11(1) : 7.

[9] 毛栊哗，李彬华，张西亮，等. 基于2m级大口径望远镜的幸运成像算法的实验研究[J]. 光学技术, 2018,44(05):542-548.
MAO L H，LI B U，ZHANG X L，et al. Experimental investigation of lucky imaging algorithm based on 2 m astronomical telescope [J]. Optical Technique，2018，44(5):542－548.

[10] 赵盼孜，李彬华，毛栊哗，等. 基于现场可编程门阵列的幸运成像算法的实现[J]. 天文研究与技术，2019，16(02):236-243.
ZHAO P Z，LI B H，MAO L H，et al. Implementation of lucky imaging algorithm based on FPGA [J]. Astronomical Research＆Technology，2019，16(2) : 236－243.

FPGA-based Lucky Imaging Algorithm in Frequency Domain

Huang Xueming[1]，Li Binhua[1,2]，Wang Jinliang[1]

(1. Faculty of Information Engineering and Automation, Kunming University of Science and Technology, Kunming 650500, China;

2. Key Laboratory of Applications of Computer Technologies of the Yunnan Province, Kunming University of Science and Technology, Kunming 650500, China, Email; lbh@bao.ac.cn)

Absrtact: Lucky imaging is a post-processing image restoration technique , which can effectively reduce the impact of atmospheric turbulence on image quality. The traditional frequency-domain lucky imaging algorithm is better than the traditional space-domain lucky imaging algorithm in imaging results. It mainly consists of three parts: image preprocessing, data selection and data superposition. This paper presented a FPGA-based lucky imaging algorithm in frequency domain, and built an experimental system on FPGA. This system implements two-dimensional fast Fourier transform(2D-FFT) on astronomical image data by converting two-dimensional to two one-dimensional processing methods; And the sum of the square of the real part and the square of the imaginary part of the frequency data is used to replace the amplitude for data selection. The obtained high-resolution image reconstructed by the FPGA system is basically the same with the image processed by the CPU platform. The experimental system can process 2000 frames of 128×128 pixels image in frequency domain, what's more, the processing speed of the system on FPGA is 13 times faster than that on CPU. It accelerates the traditional frequency-domain lucky imaging algorithm while provides a feasible technical scheme for the real-time frequency-domain lucky imaging techniques in the future.

Key words: Lucky imaging; Atmospheric turbulence; 2D-FFT; Frequency domain; FPGA